AMENDMENTS TO THE CLAIMS

1. (Original) In a data processing system having a least one processor for running tasks and resources available to tasks, a method comprising the steps of:

logically partitioning tasks into groups of tasks that utilize the same resources;

preemptively scheduling the groups of tasks to be run such that each group of tasks is given a time slot in a cycle in which its tasks may run on the processor; and

for each group, non-preemptively scheduling tasks to be run within each group during the time slot allocated to the group.

2. (Original) The method recited in Claim 1 wherein the data processing system includes at least one storage device and the method further comprises the step of storing a group list for each associated group in the storage device, wherein each group list includes identifying information for tasks included in the associated group.

3. (Original) The method recited in Claim 1, further comprising the step of storing status information for each group indicating whether the group has a task that is running and holding identifying information about any task that is running.

4. (Original) The method recited in Claim 1 wherein the step of logically partitioning tasks into groups of tasks that utilize the same resources further comprises the steps of:

initially placing each task in its own group; and

subsequently combining groups of tasks into a merged group wherein each task in the merged group allocates a common resource when run.

5. (Original) In a data processing system having at least one storage device for storing modules of code, at least one processor for running tasks wherein running each task

LAW OFFICES OF
CHRISTENSEN O'CONNOR JOHNSON KINDNESS PLLC
1420 Fifth Avenue
Suite 2800
Seattle, Washington 98101
206.682.8100

involves allocating at least one resource and resources that may be allocated to tasks, a method comprising the steps of:

providing a task dependency list for each task, said task dependency list listing resources that are candidates to be allocated when the task is run on the processor;

examining the task dependency lists to logically partition the tasks into groups of interdependent tasks;

preemptively scheduling the groups of tasks to be run such that each group of tasks is given a time slot in a cycle in which its tasks may run on the processor; and

for each group of tasks, non-preemptively scheduling tasks to be run within each group during the time slot allocated to the group.

6. (Original) The method recited in Claim 5, further comprising the step of storing a group list for each group that holds identifying information for tasks included in the group.

7. (Original) The method recited in Claim 5, further comprising the step of storing status information for each group indicating the group has a task that is running and holding identifying information about any task that is running.

8. (Original) In a data processing system having at least one storage device for storing modules of code and at least one processor for running tasks, and resources that may be allocated by tasks, wherein running each task involves allocating at least one resource, a method comprising the steps of:

providing a module and resource dependency list for each associated module of code, wherein each module and resource dependency list lists interdependent modules of code of the associated module of code and resources allocated by the associated module of code;

LAW OFFICES OF
CHRISTENSEN O'CONNOR JOHNSON KINDNESS PLLC
1420 Fifth Avenue
Suite 2800
Seattle, Washington 98101
206.682.8100

generating a task dependency list for each task by taking a logical union of modules and resources listed in the module and resource dependency lists of modules and resources that are candidates to be called when the task is run on the processor;

examining the task dependency lists to logically partition the tasks into groups of interdependent tasks; and

preemptively scheduling the groups of tasks to be run such that each group of tasks is given a time slot in a cycle in which its tasks may run on the processor; and

for each group of tasks, non-preemptively scheduling tasks to be run within each group during the time slot allocated to the group.

9. (Original) The method recited in Claim 8, further comprising the step of storing a group list for each group that holds identifying information for tasks included in the group.

10. (Original) The method recited in Claim 8, further comprising the step of storing status information for each group indicating the group has a task that is running and holding identifying information about any task that is running.

11. (Original) A data processing system, comprising:

a partitioning mechanism for partitioning tasks into groups of interdependent tasks that utilize the same resources;

an execution mechanism for executing the tasks;

a preemptive scheduler for preemptively scheduling the groups of tasks such that each group is given a time slot in which to execute one of its tasks; and

a non-preemptive scheduler for non-preemptively scheduling tasks within each group.

12. (Currently amended) In a data processing system having at least one processor for running tasks and resources that may be allocated by said tasks, wherein said processor runs an

LAW OFFICES OF
CHRISTENSEN O'CONNOR JOHNSON KINDNESS PLLC
1420 Fifth Avenue
Suite 2800
Seattle, Washington 98101
206.682.8100

operating system, a computer-readable storage medium holding the operating system, said operating system performing the steps of:

logically partitioning tasks into groups of interdependent tasks, wherein said tasks are to be run non-preemptively, said tasks allocating the same resources, and tasks that do not allocate the same resources being placed as separate groups;

preemptively scheduling the groups of tasks to be run such that each group of tasks is given a time slot in a cycle in which its tasks may run on the processor; and

for each group, non-preemptively scheduling tasks to be run within each group during the time slot allocated to the group.

13. (Original) The computer-readable storage medium of Claim 12 wherein the operating system further performs the steps of:

initially placing each task in its own group; and

subsequently combining groups of tasks into a merged group wherein each task in the merged group allocates a common resource.

14. (Original) In a data processing system having at least one storage device for storing modules of code, system resources that may be allocated by tasks, and at least one processor for running tasks wherein running each task involves allocating at least one resource, a computer-readable storage medium holding an operating system for performing the steps of:

providing a task dependency list for each task, said task dependency list listing resources that are allocated when the task is run on the processor;

examining the task dependency lists to logically partition the tasks into groups of interdependent tasks such that, for each task, modules on the task dependency list for the task are included in a single group;

LAW OFFICES OF
CHRISTENSEN O'CONNOR JOHNSON KINDNESS PLLC
1420 Fifth Avenue
Suite 2800
Seattle, Washington 98101
206.682.8100

preemptively scheduling the groups of tasks to be run such that each group of tasks is given a time slot in a cycle in which its tasks may run on the processor; and

for each group of tasks, non-preemptively scheduling tasks to be run within each group during the time slot allocated to the group.

15. (Original) In a data processing system having at least one storage device for storing modules of code, system resources that may be allocated by tasks, and at least one processor for running tasks wherein running each task involves allocating at least one resource, a computer-readable storage medium holding an operating system for performing the steps of:

providing a module and resource dependency list for each associated module of code, wherein each module and resource dependency list lists interdependent resources of the associated module of code;

generating a task dependency list for each task by taking a logical union of resources listed in the module and resource dependency lists of resources that are allocated when the task is run on the processor;

examining the task dependency lists to logically partition the tasks into groups of interdependent tasks;

preemptively scheduling the groups of tasks to be run such that each group of tasks is given a time slot in a cycle in which its tasks may run on the processor; and

for each group of tasks, non-preemptively scheduling tasks to be run within each group during the time slot allocated to the group.

LAW OFFICES OF
CHRISTENSEN O'CONNOR JOHNSON KINDNESS PLLC
1420 Fifth Avenue
Suite 2800
Seattle, Washington 98101
206.682.8100